The While statement causes WILMA to repeat an action as long as a certain condition is true.

User Interaction Statements

The User Interaction category consists of statements that allow interaction with a running macro.

Alert (*Title, Message*) — Causes the macro to stop running and displays an alert panel with a custom title, message, and two standard buttons: OK and Cancel. Once the user responds to the alert panel, the macro continues running.

Pause ( ) — Causes a macro to stop running and displays the Pause panel, with a standard message *Click Continue to continue running the macro* and a Continue button. Once the user responds to the Pause panel, the macro continues running.

Prompt (*Title, Message, Button1* [, *Button2* [, *Button3*]]) — Causes a macro to stop running and displays a Prompt panel with a custom title and message. The Prompt panel can have up to three custom buttons. Prompt returns a value associated with the button used to dismiss the panel, and this information can then be used in calculations.

Return ([*Value*]) — Stops a running macro. If the macro was called by another macro or function, the Return statement can optionally pass a value back to the calling macro or function.

Decision Control Statements

The *Decision Control* category consists of statements that interrupt or deviate from the sequential order in which macro instructions typically execute.

Branch (*Label*) — Stems to a specified label in the current macro. WILMA continues to execute macro instructions at the new location until it reaches the end of the macro.

If (*Condition*)
IfClause
[*Else*
***ElseClause*]**

The If statement is part of the If … Else structure, in which WILMA completes the instructions in the IfClause when the attached If condition is true, and skips the Else instructions. When the If condition is *not* true, WILMA completes the instructions in the ElseClause.

Loop Control Statements

The *Loop Control* category consists of statements that control the timing and order of macro instructions.

For ([*AssignList*]; [*Condition*]; [*ControlList*])
{ *Instructions* }

The For statement is a structure made of three expressions separated by semicolons. It causes WILMA to repeat an action a specified number of times. The first expression initializes one or more variables. The second consists of one or more test conditions for the variables. The third controls the value of variables while the For structure loops. Instructions in the For structure repeat until the condition being tested is no longer true.

While (Condition)
WhileClause

Info ( ) — Causes the ParaSheet Info panel to appear. This command is equivalent to the *Info Panel* command in the Info menu.

Prefs ( ) — Displays the ParaSheet Preferences panel, making its options available to the person running a macro. You must dismiss the panel for the macro to continue running. This command is equivalent to the *Preferences* command in the Info menu.

Print ( ) — Brings up the Print panel. You must dismiss the panel for the macro to continue running. This command is equivalent to the *Print* command in the main menu.

PrintNow ([*Filename*]) — Activates the specified spreadsheet and causes one copy to print, bypassing the Print panel. This command is equivalent to printing with default settings in the Print panel (the *Print* command in the main menu).

Using Statements

Before using WILMA statements, read the "About Statements" section on page 8 of this manual, which describes:

- What a statement is
- The types of statements in WILMA
- What you can do with statements

WILMA has four types of statements: Assignment, Decision Control, Loop Control, and User Interaction. All statements appear in the Statements category in the Functions panel.

Assignment Statements

The *Assignment* category consists of statements that help you represent data values using variables, and return data values from a macro.

Global (*VariableList*) — Declares and optionally assigns values to one or more global variables. Global variables are accessible from all ParaSheet spreadsheets and macros. If you don't assign a value to a global variable, it is initialized to the integer value 0. A global variable exists until you delete it.

Local (*VariableList*) — Declares and optionally assigns values to one or more local variables; that is, variables that can only be accessed from one macro. If you don't assign a value to a local variable, it is initialized to the integer value 0. A local variable exists only while the macro in which it is declared is running.

GetObjPosition (*WhichCoordinate* [, *ObjectID*]) — Returns the spreadsheet location of a chart object relative to the top and left edges of the spreadsheet.

GetObjSize (*WhichCoordinate* [, *ObjectID*]) — Returns the width or height of a chart object.

GetPageNum ([*Selection*]) — Returns either the page number on which the top left corner of a selected range of cells or object is located, or the current page number.

GetPieMarks (*WhichAttribute* [, *PieChartID*]) — Returns which labels mark each slice of a pie chart. This command corresponds to settings in the Pie Labels area of the Marking submenu in the Chart Inspector.

GetPieStartAngle ([*PieChartID*]) — Returns the starting angle of the first slice (the first plotted value) of a pie chart. This command corresponds to the Starting Angle setting in the View Options subpanel in the Chart Inspector.

GetRange ([*Selection*]) — Returns the selected range's name if one exists.

GetRowNum ([*Selection*]) — Returns the index number of the top row in a range of cells or under an object.

GetSelection ( ) — Returns the range name, object name or object number of a selection.

GetSize (*WhichAttribute* [, *Selection*]) — Returns the height of a row, or width of a column.

GetUDFCol ( ) — Returns a positive integer representing the horizontal coordinate of the cell from which a user-defined function was running. (1 represents column A, 2 represents B, and so on.)

GetUDFRow ( ) — Returns a positive integer representing the vertical coordinate of the cell from which a user-defined function was running. (1 represents row 1, 2 represents row 2, and so on.)

GetWindowPosition (*WhichCoordinate* [, *Filename*]) — Returns the location of a spreadsheet window relative to the top left corner of the screen.

GetWindowSize (*WhichCoordinate* [, *Filename*]) — Returns the width or height of a spreadsheet window.

ParaSheet Commands

The *ParaSheet* category consists of commands that affect your interaction with ParaSheet and ParaSheet spreadsheets.

Hide (*Setting*) — Conceals ParaSheet and its windows from view in the Workspace Manager. This command is equivalent to the *Hide* command in the main menu.

GetBorderShade ([*ObjectID*]) — Returns the gray-scale border shade of an object. This command corresponds to the Border color well in the Color subpanel in the Chart Inspector.

GetBorderWidth ([*ObjectID*]) — Returns the border thickness of an object. This command corresponds to the Border Weight area of the Color subpanel in the Chart Inspector.

GetChartAttribs (*WhichAttribute* [, *ChartID*]) — Returns either a name, source of chart labels, presence of legend, or whether the graph size is frozen for the specified chart. This command corresponds to settings in the Common subpanel of the Chart Inspector.

GetColNum ([*Selection*]) — Returns the index number of the column furthest left in a range of cells, or under an object.

GetDocName ( ) — Returns the name of the active spreadsheet.

GetFillShade ([*ObjectID*]) — Returns the gray-scale shade of the area within the borders of a chart object. This command corresponds to the *Fill* color well in Color subpanel in the Chart Inspector.

GetFmtType ([*Selection*]) — Returns the format applied to the selected data. (These formats correspond to predefined and custom formats in the ParaSheet Format Inspector.)

GetFont (*WhichAttribute* [, *Selection*]) — Returns either the font family, typeface, or size of characters in a range of cells, chart text object, or of a chart label. This command corresponds to settings in the Font panel.

GetInterp ([*ChartID*]) — Returns whether interpolated values have been assigned to non-numeric cells in the data set being charted. This command corresponds to the Interpolate Non-numeric Cells check box in the View Options subpanel in the Chart Inspector. It is not applicable to high-low or pie charts.

GetNumObjs ([*Filename*]) — Returns the total number of chart objects in a spreadsheet.

GetNumPages ( ) — Returns the total number of pages that will be printed in the active spreadsheet.

GetNumParts (*WhichPart* [, *ChartID*]) — Returns the total number of a specific type of chart part.

GetObjName ([*ObjectID*]) — Returns the name of a chart object. This command corresponds to the Name setting in the Common subpanel of the Chart Inspector.

GetObjNum ([*ObjectID*]) — Returns the number of a chart object. This command corresponds to the Number setting in the Chart area of the Common subpanel in the Chart Inspector.

Show area of the View Options subpanel in the Chart Inspector. It is not applicable to pie, area, or 3D charts.

Get3DViewAngle (*WhichAngle* [, *ChartID*]) — Returns a setting for one of the viewing angles, or the apparent viewing distance for a 3D chart. This command corresponds to settings in the Angles area of the View Options subpanel in the Chart Inspector.

Get3DViewDepth (*WhichAxis* [, *ChartID*]) — Returns the viewing depth of a specified axis for a 3D chart. This command corresponds to settings in the Depth area of the View Options subpanel in the Chart Inspector. Get3DViewDepth is not applicable to wireframe charts.

GetAxisGridShow (*WhichAxis* [, *ChartID*]) — Returns whether grid lines for a given axis are displayed. This command corresponds to Grid settings in the Show area of the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

GetAxisLabels (*WhichAxis* [, *ChartID*]) — Returns the location of the labels for the given axis of a chart. This command corresponds to settings in the Axis Labels area of the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

GetAxisLabelsShow (*WhichAxis* [, *ChartID*]) — Returns whether labels for a given chart axis are displayed. This command corresponds to the Labels settings in the Show area of the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

GetAxisTicksCalc (*WhichAxis* [, *ChartID*]) — Returns whether tick marks are calculated automatically or from user-defined values. This command corresponds to the Automatic Calculation check box in the Ticks area of the Marking subpanel in the Chart Inspector.

GetAxisTicksShow (*WhichAxis* [, *ChartID*]) — Returns whether tick marks for a given chart axis are showing. This command corresponds to the Ticks settings in the Show area of the Marking subpanel in the Chart Inspector.

GetAxisTicksValues (*WhichSetting* [, *ChartID*]) — Returns information describing the settings of tick marks on the y-axis (or the x-axis for scatter charts). This command corresponds to settings in the Ticks area of the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

GetAxisXScale ([*ChartID*]) — Returns the type of scaling for the x-axis of a scatter chart. This command corresponds to settings in the Axis Scaling area of the View Options subpanel in the Chart Inspector.

GetAxisYScale ([*ChartID*]) — Returns the type of scaling for the y-axis of a chart. This command corresponds to settings in the Axis Scaling area of the View Options subpanel in the Chart Inspector. It is not applicable to pie charts.

NRows ([*Range*]) — Returns a positive integer representing the total number of spreadsheet rows in a range. If you don't provide an argument, NRows returns the number of rows in the current selection.

Offset (*Range, HOffset, VOffset*) — Returns a range reference a specified distance from the top left cell in the provided range. If you use Offset alone, the value is displayed in the cell. You can also use Offset to provide an argument to a function expecting a range reference — for example, as an argument for the GetRowNum command and Sum function.

Scroll (*Type, Direction* [, *Amount*]) — Scrolls the spreadsheet display horizontally or vertically.

Select (*What*) — Selects the contents of a cell, a range of cells, an object, or a spreadsheet window.

SelectAdd (*List*) — Selects multiple objects, ranges, or cells.

SelectAll ( ) — Selects either all spreadsheet cells, all objects in the spreadsheet, or the contents of a chart text object, depending on what is currently selected. This command is equivalent to the *Select All* command in the Edit menu.

TextSelect ({ *Where* | *BeginChar, EndChar* [, *BeginWord, EndWord* [, *ObjectID*]] }) — Specifies where to position the insertion point, or provides a starting and ending point for a text selection in a chart text object.

VIndex (*Value, Column*) — Searches from top to bottom in the column furthest left in the range specified by *Column* until it finds the specified *Value*, and returns the count of the cells searched. As it searches, VIndex counts cells starting at zero. For example, if two cells are searched to locate *Value*, VIndex returns 1.

VLookup *(Value, Column, Offset)* — Searches from top to bottom in the column furthest left in a range specified by *Column* until it finds the specified *Value*. VLookup then moves right in that row *Offset* cells and returns that location.

VLookup returns the cell reference (for example G15). ParaSheet can use this reference to display the **value** in the referenced cell, which is the usual result of VLookup. However, you can also use VLookup wherever a range reference is appropriate — for example, as an argument in the Sum function.

Get Commands

The *Get* category consists of commands that return information about an aspect of the spreadsheet.

Get2DView (*Display* [, *ChartID*]) — Returns how chart data is displayed, depending on the type of chart. This command corresponds to check boxes in the

ObjSize ***(NewWidth, NewHeight)*** — Establishes a new size for the selected chart object in the spreadsheet.

PieMarksSet (*ShowLabels, Percent, Value, Category* [, *PieChartID*]) — Determines what labels mark each slice of a pie chart. PieMarksSet is only applicable to pie charts. This command is equivalent to settings in the Pie Labels area of the Marking subpanel in the Chart Inspector.

PieStartAngleSet (*Angle* [, *PieChartID*]) — Sets the starting angle for the first slice (the first plotted value) of a pie chart. PieStartAngleSet is only applicable to pie charts. This command is equivalent to the Starting Angle setting in the View Options subpanel in the Chart Inspector.

SendToBack ([*ObjectList*]) — Places specified objects behind other objects. This command is equivalent to the *SendToBack* command in the Layout submenu.

Selection Commands

The *Selection* category consists of commands for selecting data, and parts of the spreadsheet for manipulation.

Cell (*Column, Row*) — Constructs a cell reference from a pair of column and row coordinates. Cell is particularly useful in combination with other WILMA functions and commands.

ChartSelect (*PartType, PartNum, AddPart* [, *ChartID*]) — Selects part of a chart. When used multiple times with the *AddPart* argument set to "Yes", it can select several parts of a chart.

HIndex (*Value, Range*) — Searches from left to right across the first row in the range specified by Range until it finds the specified *Value*, and returns the count of the cells searched. As it searches, HIndex counts cells starting at zero. For example, if two cells are searched to locate *Value*, HIndex returns 1.

HLookup (*Value, Range, Offset*) — Searches from left to right across the first row in the range specified by *Range* until it finds the specified *Value*. HLookup then moves down in that column *Offset* cells and returns the cell at that location.

HLookup returns the cell reference (for example D15). ParaSheet can use this reference to display the **value** in the referenced cell, which is the usual result of HLookup. However, you can also use HLookup as a range reference — for example, in the macro command Select.

NCols ([*Range*]) — Returns a positive integer representing the total number of spreadsheet columns in a range. If you don't provide an argument, NCols returns the number of columns in the current selection.

NextCell (*XDirection, YDirection*) — Selects a cell that is the specified distance from the currently selected cell.

AxisTicksShow (*ShowX, ShowY* [, *ShowZ*]) — Specifies whether tick marks for each axis of the selected chart display. This command is equivalent to settings in the Show area of the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

BorderShade (*Shade* [, *ObjectID*]) — Sets the gray-scale shade of the selected line or border of the selected object. This command is equivalent to the *Border* color well setting in the Color subpanel in the Chart Inspector.

BorderWidth (*Width* [, *ObjectID*]) — Sets the thickness of the selected line or border of the selected object. This command is equivalent to the numeric *Border Weight* setting in the Color subpanel in the Chart Inspector.

BringToFront ([*ObjectList*]) — Places the specified objects on top of other objects in the spreadsheet. This command is equivalent to the *Bring To Front* command in the Layout submenu.

ChartAttribsSet (*ChartName, LabelsFrom, ShowWhat, Freeze*) —Determines the chart name, source of chart labels, presence of the legend, and whether the graph size is frozen. This command is equivalent to settings in the Common subpanel in the Chart Inspector.

ChartNew (*ChartType* [, *Selection*]) — Creates a new chart of the specified chart type, from data in a specified range of cells. This command is equivalent to choosing a chart icon from the icon bar.

ChartTextDel ([*ChartTextID* [, *ChartID*]]) — Removes a chart text object from a chart.

ChartTextNew ([*ChartText, ChartID* [, *XCoord, YCoord*]]) — Adds a chart text object to a chart. If you don't provide any arguments, ParaSheet creates an empty chart text object. This command is equivalent to the *Add Chart Text* command in the Charts menu.

FillShade (*Shade* [, *ObjectID*]) — Sets a gray-scale shade for the area within the borders of the selected object. This command is equivalent to the Fill setting in the Color subpanel in the Chart Inspector.

InterpSet (*Interpolate* [, *ChartID*]) — Specifies whether to assign interpolated values to non-numeric cells with the data set being charted. If interpolation is off, non-numeric cells are charted as if their values were zero. This command is equivalent to the Interpolate Non-numeric Cells button in the Chart Inspector. It is not applicable to high-low or pie charts.

ObjInfo *(ObjName)* — Gives a chart object a descriptive name that you specify, and optionally protects the object from being moved or changed. This command is equivalent to the Common subpanel in the Chart Inspector).

ObjMove *(XDirection, YDirection [,Reference])* — Moves the selected chart legend, graph, or a chart text object to a new location in a chart.

Object Commands

The *Object* category consists of commands for creating, editing, deleting, and formatting objects.

2DViewSet (*Display1* [, *Display2* [, *ChartID*]]) — Specifies how data is displayed, depending on the type of chart. This command is equivalent to the check boxes in the Show area of the View Options subpanel in the Chart Inspector. It is not applicable to pie, area, or 3D charts.

3DViewAngleSet (*Rotation, Elevation, Perspective* [, *ChartID*]) —Defines the viewing angles and apparent viewing distance for a 3D chart. This command is equivalent to settings in the Angles area of the View Options subpanel in the Chart Inspector.

3DViewDepthSet (*ZDepth* [, *XDepth* [, *ChartID*]]) — Defines the viewing depth of a 3D chart. This command is equivalent to the settings in the Depth area of the View Options subpanel in the Chart Inspector.

AxisGridShow (*ShowX, ShowY* [, *ShowZ*]) — Specifies whether grid lines for an axis show behind the charted data in the selected chart. This command is equivalent to the check boxes in the Show area of the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

AxisLabelsSet (*LabelY* [, *LabelX*]) — Specifies the side of the grid on which labels and tick marks of an axis are drawn. This command is equivalent to choices in the Axis Labels area of the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

AxisLabelsShow (*ShowX, ShowY* [, *ShowZ*]) — Specifies whether labels for the axis of the selected chart display. This command is equivalent to choices in the Show area of the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

AxisScale (*YAxis* [, *XAxis*]) — Sets the type of scaling for the y-axis (and for scatter charts only, for the x-axis) of the selected chart. This command is equivalent to choices in the Axis Scaling area of the View Options subpanel in the Chart Inspector. It is not applicable to pie charts.

AxisTicksCalcAuto (*YAxis* [, *XAxis*]) — Sets or resets automatic calculation of tick marks on the y-axis (and for scatter charts only, the x-axis) of the selected chart. This command is equivalent to the Automatic Calculation check box in the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

AxisTicksCalcMan (*YAxis* [, *XAxis*]) — Specifies custom settings for tick marks on the y-axis (and for scatter charts only, the x-axis) of the selected chart. This command is equivalent to settings in the Ticks area of the Marking subpanel in the Chart Inspector. It is not applicable to pie charts.

NamedRngAdd (*RangeName* [, *Selection*]) — Gives a name to the specified range of cells. This command is equivalent to using the Add button in the Named Ranges panel (the *Named Ranges* command in the Sheet menu).

NamedRngDel (*RangeName*) — Removes the specified range name from the data with which it is associated. This command is equivalent to using the *Delete* button in the Named Ranges panel (the *Named Ranges* command in the Sheet menu).

NamedRngEdit (*OldName, NewName* [, *NewRange*]) — Changes the specified range name from its current name to a new name. This command is equivalent to using the Modify button in the Named Ranges panel (the *Named Ranges* command in the Sheet menu).

PageBreakSet (*Setting, Type* [, *Selection*]) — Sets a new page break below the selected row or to the right of the selected column, or removes an existing page break at a row or column. PageBreakSet can also remove all manual page breaks from a spreadsheet. This command is equivalent to the *Set Page Break*, *Remove Page Break*, or *Remove All Breaks* commands in the Sheet menu.

PageBreaksShow (*Setting*) — Specifies whether page breaks display. This command is equivalent to the *Show Page Breaks/Hide Page Breaks* commands in the Layout submenu.

Recalc ([*Filename*]) — Activates the specified spreadsheet and recalculates that spreadsheet according to the current recalculation options. This command is equivalent to the *Recalc* command in the Sheet menu.

RecalcAll ( ) — Recalculates all open spreadsheets. It also recalculates links between all open spreadsheets and any supporting spreadsheets directly linked to them. ParaSheet does not recalculate linked spreadsheets beyond the first level of links; that is, supporting spreadsheets not directly linked to open spreadsheets. (For more information on linked spreadsheets, see page 89 in the *Tutorial and Documentation* manual.) This command is equivalent to the *Recalc All* command in the Sheet menu.

RecalcOptions (*How* [, *CalculateBy*]) — Sets the method and order of recalculation for the spreadsheet. Recalculation options correspond to choices in the Recalculation Options panel. This command is equivalent to settings in the Recalculation Options panel (the *Recalc Options* command in the Sheet menu).

SheetSizeSet ([*What, Size1* [, *Size2*]]) — Changes the number of columns and rows in a spreadsheet. This command is equivalent to settings in the Spreadsheet Size area of the Preferences panel (the *Preferences* command in the Info menu).

SizeSet (*What, Size* [, *Selection*]) — Sets the height of a row or the width of a column (in pixels).

data in the deleted custom format is reformatted using the predefined General format. This command is equivalent to selecting a format in the Format Inspector and clicking *Delete*.

FmtTypeSet ([*Type* [, *Selection*]]) — Formats data in a range of cells or in chart labels using one of the ParaSheet predefined (or custom, if any exist) data formats. This command is equivalent to setting a format in the Format Inspector. If you don't provide any arguments, the Format Inspector appears.

FontSet ([*Family, Typeface, Size* [, *Selection*]]) — Sets the font family, typeface, and size of characters in the specified range of cells or chart text object . These settings correspond to choices in the ParaSheet Font panel. If you don't provide any arguments, the Font panel appears.

Formats ( ) — Causes the Format Inspector to appear. This command is equivalent to the *Format Inspector* command in the Format menu.

Italic (*Setting* [, *Selection*]) — Determines whether the contents of selected cells or chart text objects are italic. This command is equivalent to the *Italic/ Unitalic* commands in the Font submenu.

PageLayout ( ) — Displays the ParaSheet Page Layout panel, so you can specify page layout options. This command is equivalent to the *Page Layout* command in the Format menu.

Plain ([*Selection*]) — Makes the contents of selected cells, chart text objects, or chart labels plain, unstyled characters.

PointSize (*Setting* [, *Selection*]) — Changes the font size of the contents of a selected cell or range of cells, chart text object, or chart label. This command is equivalent to settings in the Size text box in the Font panel.

Sheet Commands

The *Sheet* category consists of commands for changing how the spreadsheet displays and automating calculations.

GoTo ([*Reference*]) — Selects the specified cell, range of cells, or object. This command is equivalent to the *Go To* command in the Sheet menu. If you don't provide any arguments, the GoTo panel appears.

GridShow (*Setting*) — Specifies whether spreadsheet grid lines show when the spreadsheet is displayed and printed. This command is equivalent to the *Show Grid* or *Hide Grid* commands in the Layout submenu.

HeadingsShow (*What, Setting*) — Specifies whether the row headings (1, 2, 3 …) and column headings (A, B, C …) show when the spreadsheet is displayed and printed. This command is equivalent to the *Show Column Heading, Hide Column Heading, Show Row Heading,* or *Hide Row Heading* commands in the Layout submenu.

be the chosen combination of values, formats, and formulas) into a new location. Any absolute, relative, or mixed cell references are updated. This command is equivalent to choices in the Paste Special panel (the *Paste Special* command in the Edit menu).

Sort ([*Options* [, *Selection*]]) — Arranges the contents of a range of records according to ParaSheet sorting order. If you provide arguments, Sort is dependent on SortKeyReset to first set a sort range. If you don't provide any arguments, the Sort panel appears. This command is equivalent to selecting a sort range in a spreadsheet and setting the Ignore Case option in the Sort panel (*Sort* in the Edit menu).

SortKeyDel (*Record*) — Resets the sorting direction for a record to Ignore. It does not change the sorting priority of records relative to one another. SortKeyDel is dependent on SortKeyReset to first set a sort range. This command is equivalent to setting a cell range to the Ignore key in the Sort panel (*Sort* in the Edit menu).

SortKeyReset (*RecordsIn , Range*) — Determines whether records to be sorted are arranged in rows or columns. It also sets all record key settings for a range to Ignore except the first, which is set to ascending order. This command is equivalent to default settings in the Sort panel (*Sort* in the Edit menu).

SortKeySet (*Record, Direction* [, *Priority*]) — Determines the sort order for a record. It also sets its sort priority relative to other records in a sort range. SortKeySet is dependent on the SortKeyReset command to first set a sort range. This command is equivalent to assigning sort direction and priority to a cell range using the Set Priority buttons and sort key settings in the Sort Panel (*Sort* in the Edit menu).

Format Commands

The *Format* category consists of commands for changing the format of spreadsheets, cell contents or objects such as chart text, and chart labels.

Align (*How* [, *Selection*]) — Sets the alignment of cell contents for a selection, or the contents of a chart text object. This command is equivalent to commands in the Alignment submenu.

Bold (*Setting* [, *Selection*]) — Determines whether the contents of selected cells, chart text objects are bold. This command is equivalent to the *Bold/Unbold* commands in the Font submenu.

FmtCustomAdd ([*Name, Pattern* [, *Selection*]]) — Lets you create a custom formatting pattern for data. This command is equivalent to settings in the Format Inspector. If you don't provide any arguments, the Format Inspector appears.

FmtCustomDel (*FormatName*) — Removes a custom format from the list in the Format Inspector. (Predefined formats cannot be deleted.) Any spreadsheet

spreadsheet, depending on what was deleted. Any absolute, relative, or mixed cell references are updated. Delete does not work with multiple selections. This command is equivalent to choices in the Delete panel (the *Delete* command in the Edit menu).

Enter (*What* [, *Selection*]) — Like the Return key, the Enter command enters data into a cell. Enter also inserts text into a chart text object, in which case it positions the I-beam immediately after the last character entered. When adding text to an existing chart text object, new text is entered at the current I-beam position. If text is selected, new text replaces the selected text. If the I-beam is not positioned, new text is entered at the beginning of the existing text.

Fill (*Direction* [, *Selection*]) — Copies data (including formulas) in a selected column or row of cells and pastes it into the remaining selected cells to the right if the selection is a column, or below if the selection is a row. This command is equivalent to the *Fill Right* and *Fill Down* commands in the Fill submenu.

IncFill ([*Direction, Type, Value, Start, Step* [, *Selection*]]) — Inserts values into the selected cells by rows or by columns, starting with a specified value and increasing or decreasing by another specified value. Values can increment or decrement in either a linear or growth fashion. If you don't provide any arguments, the Incremental Fill panel appears. This command is equivalent to choices in the Incremental Fill panel (the *Incremental Fill* command in the Fill submenu).

Insert ([*ReplaceDirection* [, *Selection*]]) — Adds cells, columns, or rows to the spreadsheet. Remaining columns move to the right and rows move down in the spreadsheet, depending on what was selected. Any absolute, relative, or mixed cell references are updated. This command is equivalent to choices in the Insert panel (the *Insert* command in the Edit menu).

Move ([*Source, Destination*]) — Takes the selected data from its current location and puts it in a new location. If you don't specify any arguments but have two separate cells or ranges selected, WILMA moves data from the first to the second selection. (Move does not work with multiple selections of more than two ranges.) Any absolute, relative, or mixed cell references are updated. This command is equivalent to the *Move* command in the Edit menu.

Paste ([*Selection*]) — Inserts the contents of the pasteboard into a cell, range of cells or chart text object. Any absolute, relative, or mixed cell references are updated. If the pasteboard contents are an object, ParaSheet pastes the object at the selection you provide, aligning the top left corner of the object with the top left corner of the selection. The pasted object is given a new object number; if it was named, you must rename it if you want to refer to it by name. This command is equivalent to the *Paste* command in the Edit menu.

PasteSpecial ([*Formats, Contents, Effect, SelectedCells, AbsoluteRef* [, *Selection*]]) — Inserts the specified contents of a range of cells (which can

Save ([*Filename*]) — Stores changes made to the active spreadsheet since that spreadsheet was last saved. This command is equivalent to the *Save* command in the Document menu.

SaveAs ([*Filename*]) — Stores the active spreadsheet under another name and/or in another location. This command is equivalent to the *Save As* command in the Document menu.

SaveTo ([*Filename, FileType*]) — Prepares the active spreadsheet for use in another application or on another computer. If you don't provide any arguments, the Save To panel appears. This command is equivalent to choices in the Save To panel (the *Save To* command in the Document menu).

WindowMove (*XDirection, YDirection* [, *How*]) — Moves a selected spreadsheet window to a new location in the workspace.

WindowSize (*NewWidth, NewHeight*) — Resizes a selected spreadsheet window in the workspace.

Zoom (*Size*) — Changes the percentage of magnification at which the contents of a spreadsheet window are displayed. Zoom is only applicable to the spreadsheet that is the main window. This command is equivalent to Zoom settings at the bottom right corner of a spreadsheet window.

Edit Commands

The *Edit* category consists of commands for modifying and manipulating the contents of cells.

Clear ([*Selection*]) — Deletes the contents of a cell, range of cells, an object or multiple objects, or selected text in a chart text object. This command is equivalent to the *Clear* command in the Edit menu.

ClearSpecial ([*What* [, *Selection*]]) — Deletes the specified contents of a range of cells, which can be data, formats, or formulas. This command is equivalent to the *Clear Special* command in the Edit menu.

Copy ([*Selection*]) — Duplicates the formatting attributes and contents (including data and formulas) of a range of cells, objects (including attributes), text (including attributes) in chart text objects, or chart labels, and places the duplicated information in the pasteboard. This command is equivalent to the *Copy* command in the Edit menu.

Cut ([*Selection*]) — Removes an object or multiple objects, or the formatting attributes and contents (including data and formulas) from a cell, range of cells, or chart text object, and puts the information in the pasteboard. This command is equivalent to the *Cut* command in the Edit menu.

Delete ([*ReplaceDirection* [, *Selection*]]) — Removes cells, columns, or rows from the spreadsheet. Remaining data moves to the left or up in the

section, to illustrate their relationships to one another and with ParaSheet menu commands.

Document Commands

The *Document* category consists of commands for manipulating the spreadsheet window.

Activate (*Filename*) — Makes a spreadsheet the active window. This command is equivalent to selecting a spreadsheet name from the Windows menu. The spreadsheet must be open (it can be miniaturized) before it can be activated.

Close ([*Filename*]) — Closes a spreadsheet window. This command is equivalent to the *Close* command in the Document menu. If there are any open macros for the spreadsheet, they are first automatically saved and closed.

Import ([*Filename* [, *FileType*]]) — Opens an ASCII file or spreadsheet created in an application other than ParaSheet or on a computer running something other than NEXTSTEP. This command is equivalent to choices in the Import panel (the Import command in the Document menu).

Maximize ([*Filename*]) — Redisplays a miniaturized spreadsheet as a spreadsheet window.

Miniaturize ([*Filename*]) — Causes a spreadsheet window to display as an icon in the workspace. This command is equivalent to the *Miniaturize Window* command in the Windows menu.

New ( ) — Opens a new spreadsheet window and makes it the main window. This command is equivalent to the *New* command in the Document menu.

Open ([*FirstFile* [, ... , *LastFile*]]) — Opens one or more existing spreadsheets. This command is equivalent to opening the Open panel (using the *Open* command in the Document menu) and selecting one or more files.

OpenLinks ([*Filename*]) — Makes a specified spreadsheet the main window, then brings up the Open Links panel, so you can choose a spreadsheet linked to the main spreadsheet window. If you don't specify a filename, the Open Links panel displays spreadsheets linked to the current spreadsheet. This command is equivalent to choosing the *Open Links* command from the Document menu and selecting a file from the Open Links panel.

Redraw (*Setting*) — Reduces the amount of spreadsheet redrawing that takes place after every macro action. This command is particularly useful to save redrawing time when working with large, graphically complex spreadsheets.

RevertToSaved ([*Setting*]) — Loads the last saved version of a spreadsheet, erasing the most recent unsaved changes. This command is equivalent to the *Revert to Saved* command in the Document menu.

IfErr (*Expression, TrueVal, FalseVal*) — Tests whether a specified expression evaluates to an error, and returns *TrueVal* if it does evaluate to an error, or *FalseVal* if it doesn't.

IfX (*Condition, TrueVal, FalseVal*) —Tests whether a specified condition is true or false and returns TrueVal if it's true and FalseVal if it's false. This function should not be confused with the macro statement If.

When the result of the IfX function is a cell or range, IfX returns the range reference. Typically, this reference is used to display the **value** in the referenced cell or range. However, you can also use IfX wherever a range reference is appropriate (if ranges are used in the *TrueVal* and *FalseVal* arguments)—for example, as an argument for the Sum function.

IsNull (*Expression*) — Tests whether a specified expression is a null value. If it is, IsNull returns 1 (or logical True). If it isn't, IsNull returns 0 (or logical False).

IsNum (*Expression*) — Tests whether a specified expression is a numeric value. If it is, IsNum returns 1 (or logical True). If it isn't, IsNum returns 0 (or logical False).

IsText (*Expression*) — Tests whether a specified expression is a text value. If it is, IsText returns 1 (or logical True). If it isn't, IsText returns 0 (or logical False). Error codes are not considered text values.

NA ( ) — Assigns the value !N/A (value-not-available error) to any cell in which it is entered. This function can act as a placeholder in the spreadsheet when you are unable to enter a value needed to complete a formula. Any cell with a formula dependent upon a cell that contains !N/A will display !N/A. This prevents the occurrence of a chain of inaccurate calculations through- out a spreadsheet.

True ( ) — Returns the logical value True, which is equivalent to the numeric value 1.

Using Commands

Before using WILMA commands, read the "About Commands" section on page 6 of this manual, which describes:

- What a command is
- The types of commands in WILMA
- What you can do with commands.

WILMA has eight types of commands: Document, Edit, Format, Sheet, Object, Selection, Get, and ParaSheet. Each of these groups has its own category in the Functions panel. Commands are listed by category in this

ASC (*Character*) — Returns the ASCII value representing the specified character in the NEXTSTEP character set. ASC is the inverse of the Chr function.

Chr (*Number*) — Returns the character associated with the specified ASCII value. Chr is the inverse of the ASC function.

Left (*String, NumChars*) — Returns a specified number of characters (including spaces) furthest left in a text string.

Length (*String*) — Returns the number of characters (including spaces) in a text string.

Mid (*String, StartChar, NumChars*) — Returns a specified number of characters (including spaces) from a text string, beginning with the character in a specified position.

Num2C (*Number* [, *Format* [, *DecPlaces* [, *Commas*]]]) — Converts the numeric value of a specified number into a text string with the specified format and number, decimal places, and commas. Num2C is useful for entering a computed value as part of a text string in a cell.

Right (*String, NumChars*) — Returns a specified number of characters (including spaces) furthest right in a text string.

Logical Functions

The *Logical* category consists of functions that perform logical comparisons, logical assignments, and branching.

Choose (*n, List*) — Returns the *n*th item in a list of arguments. If the *n*th item is a cell or range, Choose returns the cell or range reference. Typically, this reference is used to display the value in the reference cell or range. You can also use Choose wherever a range reference is appropriate—for example, as an argument for the GetColNum command and Sum function.

Err ( ) — Returns the value !ERR, a ParaSheet error code. This function is equivalent to ERROR(10). You can use the Err function to identify data entry errors and other user errors in a spreadsheet, such as possible out-of-range values. Any formula that refers to a cell containing a value of !ERR also gives the value !ERR. This prevents the occurrence of a chain of inaccurate calculations throughout a spreadsheet.

Error (*Number*) — Returns the cell error code corresponding to a specified number. You can use the Error function with the IfX function to determine whether a cell contains a specific type of error, but only if the Error function is referred to indirectly by a cell reference.

False ( ) — Returns the logical value False, which is equivalent to the numeric value 0.

enter a date outside the range of valid date numbers (0 to 65534.99999), ParaSheet returns !NUM.

Day (*DNumber*) — Returns an integer from 1 to 31 (inclusive), corresponding to the day of the month for the date number you specify.

DOW (*DNumber*) — Returns an integer from 1 to 7 (inclusive), corresponding to the day of the week for the date number you specify. 1 is Sunday, 7 is Saturday.

Hour (*DNumber*) — Returns an integer from 0 to 23 (inclusive), corresponding to the hour of the day for the date number you specify. 0 is midnight, 12 is noon, and 23 is 11:00 PM.

Minute (*DNumber*) — Returns an integer from 0 to 59 (inclusive), corresponding to the minute of the hour for the date number you specify. 0 is on the hour, 1 is one minute after the hour, and 59 is one minute before the next hour.

Month (*DNumber*) — Returns an integer from 1 to 12 (inclusive), corresponding to the month of the year for the date number you specify. 1 represents January, 12 represents December.

Now ( ) — Returns the date number of the current date and time. The value of Now is updated every time the spreadsheet is recalculated.

Second (*DNumber*) — Returns an integer from 0 to 59 (inclusive), corresponding to the second of the minute for the date number you specify. 0 is on the minute, 1 is one second after the minute, and 59 is one second before the next minute.

Time (*Hour, Minute, Second*) — Returns the date number of the time specified by an hour, minute, and second. The returned date number is a decimal fraction from 0 (for 00:00:00 or 12:00:00 midnight) to .99999 (for 23:59:59 or 11:59:59 PM).

Time2C (*DNumber, DFormat, TFormat* [, *TimeFirst*]) — Returns the text string representing the date and time specified by a date number, using the formats specified by the date format number (*DFormat*) and the time format number (*TFormat*). Either the time or date can appear first, depending on the value of *TimeFirst*.

Year (*DNumber*) — Returns the year, an integer from 1904 to 2083 (inclusive), corresponding to the date number *DNumber*.

String Functions

The *String* category consists of functions for evaluating and manipulating text in the spreadsheet.

Floor (*Number*) — Returns the largest integer less than or equal to a specified number.

Max (*List*) — Returns the maximum of the numeric values in a list.

Min (*List*) — Returns the minimum of the numeric values in a list.

Round (*Number, NumDigits*) — Returns the value of a specified number, rounded to the specified number of decimal places.

StdDev (*List*) — Returns the sample standard deviation (using the method of least squares) of the numeric values in a specified list by taking the square root of the variance (difference). The formula used is:

Sqrt (*Variance* (*List*))

Sum (*List*) — Adds up the numeric values in a specified list.

Variance (*List*) — Calculates the sample variance of the values in a specified list, using the following formula:

$$\Sigma(X - Avg)^2 / (n - 1)$$

where each X is a value in the sample, Avg is the average of the values in the sample, and n is the number of values in the sample.

Time Functions

The *Time* functions help you create formulas based on dates and times—for example, formulas that figure elapsed time or calculate future dates. The Time functions are also helpful for displaying dates and times in cells.

Most Time functions use an argument called *DNumber*. *DNumber* represents a date number. The date number represents the days, hours, minutes, and seconds since 12:00:00 AM January 1, 1904. Each date and time has a particular date number. For example, the date number 31810 stands for February 3, 1991, 12:00:00 AM. The fractional part of a date number represents a fractional portion of a 24-hour day. In this manner, .0 (no fractional part) represents 12:00:00 AM and .99999 is 11:59:59 PM.

DNumber must be a numeric value in the range 0 to 65534.99999. The minimum value for *DNumber* (0) represents 12:00:00 AM January 1, 1904, and the maximum value (65534.99999) represents 11:59:59 PM, June 4, 2083.

Date (*Year, Month, Day*) — Returns the date number of a specified date. The date number is the ParaSheet identification number for that particular date. For more information on date numbers, see the discussion of *DNumber*, above. If you

there is an initial investment at the beginning of the first period, you must add it separately.

Pmt (*Principal, Rate, Term*) — Calculates the size of periodic payments necessary to pay off a loan at a periodic interest rate, over the number of payment periods in a specified term using the following formula:

$$Principal * (Rate / 1 - (Rate + 1)^{-Term}$$

Pmt assumes payments are made at the end of each period. If payments are made at the beginning of each period, divide the result by (1 + *Rate*).

PV (*Pmt, Rate, NumPmts*) — Calculates the present value of a series of payments (each of a specified amount) that collect a specific periodic interest rate over the specified number of payment periods. PV assumes payments are made at the end of each period. If payments are made at the beginning of each period, multiply the result by (1 + *Rate*). PV uses the formula:

$$[Pmt * (1 - ((1 + (Rate))^{-NumPmts}))] / Rate$$

SL (*Inv, Residual, NumPeriods*) — Calculates the straight-line depreciation amount per period of an initial investment, given the number of time periods over which to depreciate, and the residual value at the end of the term. The formula used is:

$$(Inv - Residual) / NumPeriods$$

SOYD (*Inv, Residual, Term, Period*) — Returns the depreciation amount for an investment for a given period, using a sum of the years' digits depreciation. This function uses the original investment, the number of years over which to depreciate, the residual value at the end of the term, and the period for which the depreciation amount is to be calculated. ParaSheet uses the following formula for SOYD:

$$[2 * (1 + Term - Period) * (Inv - Residual)] / [Term * (Term + 1)]$$

Statistical Functions

The *Statistical* category consists of functions for statistical operations and analysis, such as calculating averages, variance, minima, and maxima.

Abs (*Number*) — Gives the absolute value, or positive form, of a specified number.

Avg (*List*) — Returns the average (mean) of the numeric values in a list.

Ceiling (*Number*) — Returns the smallest integer greater than or equal to a specified number.

Count (*List*) — Returns the total number of numeric values in a specified list.

Tan (*Angle*) — Returns the tangent of a specified angle, given in radians. To convert a degree value to radians, use the D2R function or multiply it by pi/180.

Financial Functions

The *Financial* category consists of functions for calculations dealing with money, such as loan payments, interest, and present and future values.

Annuity (*Rate, NumPeriods*) — Returns the amount of money payable at a regular interval. The annuity is calculated from the rate and number of payment periods, according to the following formula:

$$(1 - (1 + Rate)^{-\mathrm{NumPeriods}}) / Rate$$

Compound (*Rate, NumPeriods*) — Returns the amount of interest being paid on both the interest accrued and the principal, according to the following formula:

$$(1 + Rate)^{\mathrm{NumPeriods}}$$

DB (*Inv, Term, Mult, Period*) — Returns the amount an original investment has decreased in value in a specified period. The formula ParaSheet uses to calculate the declining balance depreciation is:

$$Inv * Mult / Term * (1 - Mult / Term)^{(\mathrm{Period} - 1)}$$

FV (*Pmt, Rate, NumPmts*) — Returns the future (compounded) value of an investment based on the amount of interest earned during periodic payments. FV assumes payments are made at the end of each period. If payments are made at the beginning of each period, multiply the result by (1 + *Rate*). ParaSheet calculates the future value according to the following formula:

$$[Pmt * (((1 + Rate)^{\mathrm{NumPmts}}) - 1)] / Rate$$

IRR (*Estimate, Flows*) — Returns an approximate internal rate of return (the amount yielded) for a series of cash flows, such that the net present value of the cash flows is 0. The internal rate of return is calculated iteratively to 8-digit precision to achieve the rate for which

$$\Sigma[f_i / (1 + Rate)^i] = 0$$

where the fi represents the specified cash flows. ParaSheet begins computing the internal rate of return as the value of a specified estimate. It stops after achieving 8-digit precision, or after 20 iterations.

NPV (*Rate, Flows*) — Returns the net present value of future cash flows, discounted at a periodic interest rate, using the formula:

$$\Sigma[f_i / (1 + Rate)^i]$$

where the f_i are the cash flows. ParaSheet assumes that the cash flows occur at equal time intervals, with the first flow occurring at the end of the first period. If

Frac (*Number*) — Returns the absolute (positive) value of the fractional portion of a specified number.

Int (*Number*) — Returns the integer part of a specified number. Int rounds up for negative numbers and down for positive numbers.

Ln (*Number*) — Returns the natural logarithm, or log base *e*, of a specified number. Ln is the inverse of the Exp function.

Log (*Number*) — Returns the common logarithm, or log base 10, of a specified number.

Mod (*Number, Base*) — Returns the remainder of a specified number after it has been divided by a base number, according to the following formula:

Number – Floor *(Number / Base)* * *Base*

where Floor *(Number / Base)* is the largest integer <= the value of *(Number / Base)*.

The returned integer will be between 0 and the base number, excluding the base number.

Pi ( ) — Returns the value of pi (to a maximum of 14 digits), 3.1415926535897, which is the ratio of the circumference of a circle to its diameter. To convert degrees into radians for the trigonometric functions, you can use the D2R function or multiply the angle in degrees by pi/180.

R2D (*Number*) — Converts a number from radians to degrees, according to the following formula:

Number * 180 / p

R2D is particularly useful with mathematical functions, such as Acos, Asin, Atan, Atan2, Cos, Cot, Sin, and Tan.

Random ([*Maximum*]) — Returns a random rational number between 0 and 1, including 0 but not 1. If you specify an argument, Random returns a random number between 0 and *Maximum*, including 0, but not including *Maximum*. The results of the Random function are different with each use, and change each time ParaSheet completes a recalculation.

Sign (*Number*) — Returns whether a specified number is positive, zero, or negative.

Sin (*Angle*) — Returns the sine of an angle given in radians. Its value is always between –1 and 1, inclusive. To convert a degree value to radians, use the D2R function or multiply it by pi/180.

Sqrt (*Number*) — Returns the positive square root of a specified number.

Sum (*List*) — Adds up the numeric values in a specified list.

WILMA has six types of functions: mathematical, financial, statistical, time, string, and logical. Each of these groups has its own category in the Functions panel.

Math Functions

The *Math* category consists of functions for mathematical operations. Trigonometric, logarithmic, and other mathematical functions are included in this category.

Acos (*Number*) — Returns the arccosine, in radians, of a specified number (which is the cosine of an angle). Its value is always between 0 and +pi, inclusive. To convert the result from radians to a degree value, use the R2D function or multiply it by 180/pi.

Asin (*Number*) — Returns the arcsine, in radians, of a specified number (which is the sine of an angle). Its value is always between –pi/2 and +pi/2, inclusive (in quadrant IV or I). To convert the result from radians to a degree value, use the R2D function or multiply it by 180/pi.

Atan (*Number*) — Returns the arctangent, in radians, of a specified number (which is the tangent of an angle). Its value is always between –pi/2 and +pi/2, inclusive (in quadrant IV or I). To convert the result from radians to a degree value, use the R2D function or multiply it by 180/pi.

Atan2 (*XNumber, YNumber*) — Returns the arctangent, in radians, of two specified numbers (which are coordinates determining an angle). Its value is always between –pi and +pi, inclusive. To convert the result from radians to a degree value, use the R2D function or multiply it by 180/pi.

Cos (*Angle*) — Returns the cosine of an angle given in radians. Its value is always between –1 and 1, inclusive. To convert a degree value to radians, use the D2R function or multiply it by pi/180.

Cot (*Angle*) — Returns the cotangent of an angle given in radians. To convert a degree value to radians, use the D2R function or multiply it by pi/180.

D2R (*Number*) — Converts a number from degrees to radians, according to the following formula:

Number * p / 180

D2R is particularly useful with mathematical functions, such as Acos, Asin, Atan, Atan2, Cos, Cot, Sin, and Tan.

Exp (*Number*) — Returns the number *e* (approximately 2.7182818) raised to the power of a specified number. *e* is the constant used as the base in a natural logarithm. To calculate powers of other bases, use the exponentiation operator (^). Exp is the inverse of the Ln function.

How to Use This Reference

This chapter lists WILMA functions, commands, and statements in two different ways. The first part provides a categorical list; the second an alphabetical directory for the entire language. The categorical list is organized the same way as the Functions panel in the Tools menu.

If you have a task in mind for which you need a function, command, or statement but aren't sure where to look, try browsing the categorical lists in this chapter. First, consider what kind of task you want to complete, and find an appropriate category in one of the lists. Then, browse each command in the list, reading the command descriptions until you find the correct match. Finally, enter the syntax as shown in the categorical list; or if you need more information, refer to the alphabetical directory for the correct argument entries and a more detailed description.

If you know exactly which function, command, or statement to use but aren't sure how to use it, look it up in the alphabetical directory. Each function, command, and statement is listed with the correct syntax, arguments, a complete definition, and acceptable entries. For functions or commands that return something complex or not obvious, the possibilities are described in a table. At the end of the description are examples of how to use the command, function, or statement.

Using Functions

Before using WILMA functions, read Chapter 7 in the *Tutorial and Documentation* manual, which describes:

- What a function is
- The types of functions in WILMA
- The parts of a function
- How to use functions in your spreadsheets

It also gives information on building formulas with functions, and describes the ParaSheet mathematical and logical operators.

Once you have a working familiarity with spreadsheet functions, this reference will give you the detailed information you need to use functions effectively in ParaSheet.

Unless stated otherwise, the following conventions apply to all descriptions of functions, commands, and statements in this reference:

- Arguments surrounded by square brackets ([ ]) are optional.
- Argument entries shown surrounded by quotation marks must be entered with the quotation marks. Object names, filenames, and path names must also be surrounded by quotation marks (or assigned to string variables) when used as arguments.
- When using commands that retrieve unopened files stored in a folder other than the current folder, you must include the path name to that folder. If you use a path name in an argument, the working folder is changed to that path.
- For arguments that accept "Yes" or "No" entries, you can enter "True" or 1 instead of "Yes", or "False" or 0 instead of "No".
- Variable names, labels, filenames, path names, and macro names are case sensitive. Object names, range names, and text equivalents to numeric arguments are *not* case sensitive.
- Get commands almost always return only one specified value at a time, though their counterpart Set commands are capable of setting several values at a time. (For example, Get3DViewAngle is capable of returning settings for either the rotation, elevation or perspective of a chart, but not all at the same time. 3DViewAngleSet can set all angles at once.) If you attempt to specify multiple selections for any Get command, WILMA will return –1. (For more information, see individual Get commands beginning on page 67.)
- Examples are included for the purpose of illustrating how functions, commands, and statements can be used in formulas and macros. Some of these examples rely on data sets not provided in this manual, or do not include complete context. If you type an example of a function in one of your spreadsheets, remember to precede it with an equals sign when necessary so you get correct results.
- Some functions and commands accept a range as an argument. If the argument needs only one value, it uses the value in the top left cell of the range and ignores the remaining values in the range. (For more information, see the list of individual functions and commands beginning on page 57.)

Chapter 2

WILMA Reference

This chapter is a complete reference to WILMA (Windowing Interactive Language for Macro Applications), the ParaSheet programming language. It includes information on categories of WILMA functions, commands and statements, as well as an alphabetical directory to WILMA.

About This Reference

This reference uses certain terminology and conventions to make information about WILMA easily accessible.

Terminology and Conventions

Unless stated otherwise, the following terminology applies to all descriptions of functions, commands, and statements in this reference:

- The word *spreadsheet* is used to describe the spreadsheet that is the main window.
- The word *object* is used to describe a chart object.